UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/659,221	09/10/2003	James R. Larus	MS303259.2/MSFTP447USA	5751

27195 7590 12/15/2008

AMIN, TUROCY & CALVIN, LLP
127 Public Square
57th Floor, Key Tower
CLEVELAND, OH 44114

EXAMINER	
DAVE, JYOTI D	

ART UNIT	PAPER NUMBER
2191	

NOTIFICATION DATE	DELIVERY MODE
12/15/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

docket1@thepatentattorneys.com
hholmes@thepatentattorneys.com
lpasterchek@thepatentattorneys.com

Office Action Summary	Application No.	Applicant(s)	
	10/659,221	LARUS ET AL.	
	Examiner	Art Unit	
	JYOTI D. DAVE	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *04 December 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-63* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-63* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *10 September 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *1/10/06--5 pgs; 3/24/04 --4 pgs*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Claim Rejections - 35 USC § 101

1. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 1-37, 38-44, 58-62 and 63 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

In claim 1, a "system that facilitates asynchronous programming" comprising "a contract component" and a "conformance checking component" are representative of the software. Therefore, claim 1 has been rejected because it is reasonably interpreted as functionally descriptive material, per se. Software, per se, is not one of the statutory subject matter.

Claims 2-36 are dependent upon claim 1 and are also interpreted as functionally descriptive material.

In claims 38 and 63, a "system that facilitates asynchronous programming" comprising "client and service interfaces", "an extraction component", "a relator component" and a "conformance checking component" are representative of the software. Therefore, claim 38 has been rejected because it is reasonably interpreted as functionally descriptive material, per se. Software, per se, is not one of the statutory subject matter.

Claims 39-44 are dependent upon claim 38 and are also interpreted as functionally descriptive material.

In claim 58, a "computer software product" comprising "a contract component", "an extraction component", "a relator component" and a "conformance checking component" are representative of the software. Claim 58 recites the execution of the computer software product by a processing unit. However, this is the intended use of computer software product. Therefore, claim 58 has been rejected because it is reasonably interpreted as functionally descriptive material, per se. Software, per se, is not one of the statutory subject matter.

Claims 59-62 are dependent upon claim 58 and are also interpreted as functionally descriptive material.

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

3. Claims 1-17 and 19-63 are rejected under 35 U.S.C. 102(e) as being anticipated by Rehof et al. (2003/0204570 A1)

Claim 1. A system that facilitates asynchronous programming, comprising:

a contract component that facilitates stating a behavioral contract (see paragraph 0009, behavioral module that reflects all types of processes (behaviors) between server and client) **on an asynchronous service interface** (see Abstract, asynchronous service)**between a client interface and a service interface** (see Fig. 1, message passing between server and client)**; and**

a conformance checking component that checks that at least one of the client interface and the service interface conform to the contract defined there between (see paragraph 0009, behavioral analysis checks whether the implementation (of the message passing program module) conforms to an intended set of type processes)**.**

Claim 2. The system of claim 1, the contract defines an order in which clients of the service interface are invoked (see paragraph 0009, various behavioral properties of the message passing program module may be evaluated [the order in which clients of the service interface are invoked is a behavioral property of the message passing program])**.**

Claim 3. The system of claim 2, the checking component checks that the clients handle both normal and exceptional results (see paragraph 0032, evaluates whether the implementation performs consistently with the processes expressed by all types of annotations, communication protocols that represent the input/output actions that developer expects each operation to perform or take on [normal results]...see also, paragraph 0050, various type annotations that specify intended message passing actions for the program module [exceptional results]…see also Fig. 4, error message 416 indicative of the fact that the CCS model 406 is not a valid abstraction of the implementation of the program module).

Claim 4. The system of claim 1, at least one of the asynchronous service interface and the contract are defined according to an objected oriented program language (see paragraph 0025, In this embodiment, as one example, the server program module 102 provides a user with a graphical user interface (GUI)[object oriented program language).

Claim 5. The system of claim 1, the checking by the checking component is modular wherein the contract is specified on an interface boundary of the client interface and the service interface and, each of the client interface and the service interface is checked independently, and each method is checked separately (see paragraph 0027, a first message passing module (client) and a second message passing module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module]).

Claim 6. The system of claim 1, the checking component further comprises a model extraction component that automatically extracts a behavior model that is fed to a model checker component (see paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action to be performed or taken by each operation, and evaluating whether the implementation performs consistently with the process expressed).

Claim 7. The system of claim 6, the extracted model is sound in the presence of aliasing (see paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action to be performed or taken by each operation).

Claim 8. The system of claim 1, further comprising a construct in the form of an asynchronous call that returns a future-like handle to an eventual result of the call (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels).

Claim 9. The system of claim 1, further comprising a construct in the form of a synch statement (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels, see also Abstract, an asynchronous message program [the input/output would have be to a synch statement because it is modeling an asynchronous message program).

Claim 10. The system of claim 1, further comprising a construct that states the behavioral contract on the service interface (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels).

Claim 11. The system of claim 1, further comprising a construct that is a transaction directive that delineates the scope of conversations by clients (see paragraph 0010).

Claim 12. The system of claim 1, further comprising a construct that is a tracked type qualifier used by the checking component (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels…see paragraph 0011, using the type annotations to check behavioral properties).

Claim 13. The system of claim 1, further comprising a procedure, which procedure is a method that is called at least one of synchronously and asynchronously (see fig. 1, server/client call operations [procedures] asynchronously (see Abstract)).

Claim 14. The system of claim 13, the asynchronous call executes concurrently with its caller by completing immediately, while the method that is invoked executes (see paragraph 0003 and Abstract).

Claim 15. The system of claim 14, the asynchronous call completes by returning a value (see paragraph 0003).

Claim 16. The system of claim 15, the value is used explicitly to synchronize the asynchronous call with the value (see paragraph 0003, Software programming has changed over the years as computer programs have moved away from a sequential model of performing operations and toward a more asynchronous model wherein multiple operations of a single program may be performed substantially simultaneously. These computer programs are typically referred to as "event-driven" programs in that a user may initiate an event in the program and thereafter fail to take any other action with respect to that event for an indefinite period of time. As such, the program is in a waiting pattern in which the program takes no action with respect to

this event until the user initiates such. During this interim, the user may initiate any number of other events in the computer program, take any number of other actions with respect to these other events, or perform no action during this interim).

Claim 17. The system of claim 15, the value is a first-class type that can be at least one of stored, passed as an argument, and returned as a result (see Fig. 1, server and client are exchanging/storing data from the operations...see also, paragraph 0010, input/output action that the program module expects each operation to perform).

Claim 19. The system of claim 15, the value is an object in one of three states that include an undefined state, normal state, and exceptional state (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect [expect undefined, normal or exception value] each operation to perform or take on a selected set of communication channels).

Claim 20. The system of claim 1, the contract is a source-level contract that facilitates statically detecting asynchronous message passing errors (see Abstract, A system and method for modeling a message-passing program module using type annotations is disclosed. The message-passing program module is constructed with operations that communicate with

operations of other message-passing program modules in an asynchronous computing environment...see also, Fig. 4, element 416, detecting errors) .

Claim 21. The system of claim 1, the contract expresses stateful protocols that govern interactions between a client associated with the client interface and a service associated with the service interface (see paragraph 0047, FIG. 3 shows a concurrent software system 300 wherein two program modules--a sender program module 304 and a receiver program module 302--communicate with one another as the program modules 304 and 302 are implemented in an asynchronous computing environment 100 in accordance with an embodiment of the present invention...see also, paragraph 0009, various behavioral properties of the message passing program module may be evaluated [the order in which clients of the service interface are invoked is a behavioral property of the message passing program]).

Claim 22. The system of claim 1, the service interface is associated with a service contract whose language models the effects of multiple clients that concurrently access the service interface (see Abstract, message-passing program modules in an asynchronous computing environment).

Claim 23. The system of claim 1, the checking component checks that services and clients obey the contract associated with the service interface (see paragraph 009, The behavioral

analysis system includes a type system that receives an implementation for the message-passing program module and thereafter checks whether the implementation conforms to an intended set of type processes).

Claim 24. The system of claim 1, the checking component uses a transaction directive to delimit a scope of an instance of a concurrent interaction between a client and a set of service instances (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels).

Claim 25. The system of claim 24, the transaction directive specifies a correlation variable that uniquely identifies the instance of the concurrent interaction between the client and the set of service instances (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels).

Claim 26. The system of claim 1, the checking component is modular in that to check for the client conformance and the service conformance, only the associated contract needs to be referenced (see paragraph 009, The behavioral analysis system includes a type system that

receives an implementation for the message-passing program module and thereafter checks whether the implementation conforms to an intended set of type processes).

Claim 27. The system of claim 1, the checking component considers a method as a function of an object to which it belongs (see paragraph 0027, a first message passing module (client) and a second message passing module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module]).

Claim 28. The system of claim 1, the checking component relates a contract state, a client state, and a service state (see paragraph 0027, a first message passing module (client) and a second message passing module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module]).

Claim 29. The system of claim 28, the checking component further comprises an extraction component that extracts a client model, service model, and interface model in order to relate the contract state, client state, and service state (see paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action to be performed or taken by each operation, and evaluating whether the implementation performs consistently with the process expressed…see also, see paragraph 0027, a first message passing module (client) and a second message passing

module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module]).

Claim 30. The system of claim 28, the checking component further comprises an extraction component that uses a region to model relevant data, which relevant data is that data which is directly relevant to the contract state (see paragraph 0027, a first message passing module (client) and a second message passing module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module]…see also, see paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action to be performed or taken by each operation, and evaluating whether the implementation performs consistently with the process expressed…see also, see paragraph 0027, a first message passing module (client) and a second message passing module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module]).

Claim 31. The system of claim 30, the region is polymorphic (see paragraph 0025, Any number of instances of the selected operation or any other operation within the server program module 102 may operate concurrently with one another...see also, fig. 1).

Claim 32. The system of claim 30, the region is partial such that only relevant data is directly assigned to a region type (see fig. 1).

Claim 33. The system of claim 1, the checking component further comprises an extraction component that facilitates model reduction by utilizing only necessary statements (see paragraph 0076).

Claim 34. The system of claim 1, the checking component further comprises an extraction component that utilizes region-and-effect analysis to extract a model (see paragraph 0076-0077).

Claim 35. The system of claim 1, the checking component further comprises an extraction component that utilizes type-and-effect inference and source-level tracked types (see paragraph 0076-0077).

Claim 36. The system of claim 1, the contract is programmer-specified to separately check client and server code (see paragraph 0027, a first message passing module (client) and a second message passing module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module]).

Claim 37. A computer according to the system of claim 1 (see fig. 2)**.**

Claim 38. A system that facilitates asynchronous programming, comprising:

a client interface and a service interface (see fig. 1)**;**

an extraction component that automatically extracts a client model from the client interface and a service model from the service interface (see paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action to be performed or taken by each operation, and evaluating whether the implementation performs consistently with the process expressed)**;**

a relator component that creates a behavioral relationship between the client model and the service model (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels)**; and**

a conformance checking component that checks that the client interface obeys the behavioral relationship, and the service interface properly implements the behavioral relationship (see paragraph 0009, behavioral analysis checks whether the implementation (of the message passing program module) conforms to an intended set of type processes)**.**

Claim 39. The system of claim 38, the behavioral relationship expressed in terms of at least one of a future, a join on the future, and asynchronous function calls (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels, see also Abstract, an asynchronous message program [the input/output would have be to a synch statement because it is modeling an asynchronous message program).

Claim 40. The system of claim 38, the behavioral relationship is a contract the terms of which are enforced by the checking component (see paragraph 0009, behavioral module that reflects all types of processes (behaviors) between server and client…see also, paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action to be performed or taken by each operation, and evaluating whether the implementation performs consistently with the process expressed).

Claim 41. The system of claim 38, the checking component checks that clients of the client interface are invoked in the proper order and that the clients handle both normal and exceptional results (see paragraph 0032, evaluates whether the implementation performs consistently with the processes expressed by all types of annotations, communication protocols that represent the input/output actions that developer expects each operation to perform or take

on [normal results]...see also, paragraph 0050, various type annotations that specify intended message passing actions for the program module [exceptional results]…see also Fig. 4, error message 416 indicative of the fact that the CCS model 406 is not a valid abstraction of the implementation of the program module).

Claim 42. The system of claim 38, the extraction component creates the client model that defines relevant data for interaction with the service interface (see paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action [interaction with service interface] to be performed or taken by each operation, and evaluating whether the implementation performs consistently with the process expressed).

Claim 43. The system of claim 38, the extraction component creates the service model that defines relevant data for interaction with the client interface (see paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action [interaction with service interface] to be performed or taken by each operation, and evaluating whether the implementation performs consistently with the process expressed).

Claim 44. The system of claim 38, the relator component checks the client model and the service model such that contract descriptions of the behavioral relationship are defined between the client model and the service model (see paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action [interaction with service interface] to be performed or taken by each operation, and evaluating whether the implementation performs consistently with the process expressed)**.**

Claim 45. Claim 45 is rejected on the same basis as claim 38.

Claim 46. Claim 46 is rejected on the same basis as claim 39.

Claim 47. Claim 47 is rejected on the same basis as claim 40.

Claim 48. Claim 48 is rejected on the same basis as claim 41.

Claim 49. The method of claim 45, the further comprising extracting with a model extraction component at least one of a client model for the client interface and a service model for the service interface (see paragraph 0027, a first message passing module (client) and a second message passing module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module])**.**

Claim 50. Claim 50 is rejected on the same basis as claim 43.

Claim 51. Claim 51 is rejected on the same basis as claim 42.

Claim 52. Claim 52 is rejected on the same basis as claim 44.

Claim 53. The method of claim 45, further comprising automatically extracting the behavioral relationship (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels).

Claim 54. A method of asynchronous programming, comprising:

receiving a client interface and a service interface (see fig. 1);

automatically extracting a client model from the client interface and a service model from the service interface (see paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action to be performed or taken by each operation, and evaluating whether the implementation performs consistently with the process expressed);

creating a behavioral relationship between the client model and the service model (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels); **and**

enforcing the behavioral relationship between the client interface and the service interface (see paragraph 0009, behavioral analysis checks [enforcing the behavioral relationship], whether the implementation (of the message passing program module) conforms to an intended set of type processes)**.**

Claim 55. The method of claim 54, the behavioral relationship is a contract expressed by constructs in terms of at least one of a future, a join on the future, and asynchronous function calls, and the terms (see paragraph 0010, type annotations are communication protocols that represent processes of input/output actions that the program module developers expect each operation to perform or take on a selected set of communication channels, see also Abstract, an asynchronous message program [the input/output would have be to a synch statement because it is modeling an asynchronous message program) **of which are enforced by a modular checking component** (see paragraph 0027, a first message passing module (client) and a second message passing module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module])**.**

Claim 56. Claim 56 is rejected on the same basis as claim 41.

Claim 57. Claim 57 is rejected on the same basis as claim 38.

Claim 58. Claim 57 is rejected on the same basis as claim 38.

Claim 59. The product of claim 58, the client model, service model, and interface model contain only respective communication actions and operations that are relevant for maintaining the state of a communication protocol (see paragraphs 0076-0077).

Claim 60. The product of claim 59, the relevant operations are represented by the use of regions (see paragraph 0027, a first message passing module (client) and a second message passing module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module]…see also, see paragraph 0031 and 0032, generating [extracting] a CCS process expression defined by the program module developer that specify the intended message passing action to be performed or taken by each operation, and evaluating whether the implementation performs consistently with the process expressed…see also, see paragraph 0027, a first message passing module (client) and a second message passing module (server), see paragraph 0032, present invention checks each operation (method) of the module [emphasis on singular use module]).

Claim 61. The product of claim 58, the interface model includes one or more effect processes for each method of the interface state (see paragraph 0047, FIG. 3 shows a concurrent software system 300 wherein two program modules--a sender program module 304 and a receiver program module 302--communicate with one another as the program modules 304 and 302 are implemented in an asynchronous computing environment 100 in accordance with an embodiment of the present invention…see also, paragraph 0009, various behavioral properties of

the message passing program module may be evaluated [the order in which clients of the service interface are invoked is a behavioral property of the message passing program]).

Claim 62. The product of claim 61, the effect process models a method of the interface state according to an effect a call has on the contract state, a future created by an asynchronous call to the method, and futures passed into the method (see paragraph 0047, FIG. 3 shows a concurrent software system 300 wherein two program modules--a sender program module 304 and a receiver program module 302--communicate with one another as the program modules 304 and 302 are implemented in an asynchronous computing environment 100 in accordance with an embodiment of the present invention…see also, paragraph 0009, various behavioral properties of the message passing program module may be evaluated [the order in which clients of the service interface are invoked is a behavioral property of the message passing program]).

Claim 63. Claim 63 is rejected on the same basis as claim 38.

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over Rehof et al. (2003/0204570 A1).

Claim 18. Rehof does not specifically disclose **The system of claim 17, the result returned is explicitly retrieved through a join statement.** However, it would be obvious to include a join statement because Rehof discloses in paragraph 0003, the program is in a waiting pattern in which the program takes no action with respect to this event until the user initiates such. During this interim, the user may initiate any number of other events in the computer program, take any number of other actions with respect to these other events, or perform no action during this interim. The Join statement would allow developer to combine the data from multiple actions in multiple tables to quickly and efficiently process large quantities of data. Rehof discloses combining the data from multiple actions and efficiently process large quantities of data[taking a number of actions with respect to other events], so it would be obvious to use a Join statement because it was a commonly used statement at the time of the invention.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jyoti D. Dave whose telephone number is 571-270-1470. The examiner can normally be reached on 7:30 AM to 5 PM Mon-Fri, Alt Fri. Eastern Time.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen can be reached on 571-272-3708. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Jyoti D Dave/ 12/4/2008
Examiner, Art Unit 2191
/Wei Y Zhen/

Supervisory Patent Examiner, Art Unit 2191